Trabajo Practico 2: Git y GitHub

Alumno: Lautaro Laner

link a repositorio actividad 2: https://github.com/Lautalocos/punto-2-tp-semana-2

link a repositorio actividad 3: https://github.com/Lautalocos/conflict-excercise/tree/main

Actividades
1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

• ¿Qué es GitHub?
GitHub es un repositorio remoto donde los usuarios pueden subir y compartir su codigo como repositorios de forma publica o privada. Esta pagina permite al usuario guardar cada cambio de su proyecto y “mergear” cambios hechos en una rama a otra, y permite a otros usuarios a colaborar en los proyectos de otro o copiar el proyecto de otro para crear su propia versión.

• ¿Cómo crear un repositorio en GitHub?

Con una cuenta de GitHub ya creada, en la pagina principal tocar el botón create repository, eso abre una pagina en la cual se le pone un titulo al nuevo repositorio y opcionalmente una descripción, luego clickear en “create repository”

• ¿Cómo crear una rama en Git?

Se usa el comando git branch “nombre de rama” (si no se le pone nombre solo mostrara las ramas existentes sin crear una nueva)

• ¿Cómo cambiar a una rama en Git?

Se usa el comando git checkout “nombre de rama”

• ¿Cómo fusionar ramas en Git?

Se usa el comando git merge “nombre de rama”, eso fusiona la rama “nombre de rama” con la rama en la que se esta trabajando

• ¿Cómo crear un commit en Git?

Se usa el comando git add .

• ¿Cómo enviar un commit a GitHub?

Se usa el commando git commit -m “descripción” , es recomendable anotar los cambios hechos en descripcion

• ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de tu código guardada en internet

• ¿Cómo agregar un repositorio remoto a Git?

Se usa el comando git remote add origin “www.ejemplo.com/repositorio.git”

• ¿Cómo empujar cambios a un repositorio remoto?
Se usa el comando git push

• ¿Cómo tirar de cambios de un repositorio remoto?

Se usa el comando git pull “nombre de repositorio”

• ¿Qué es un fork de repositorio?

Un fork de un repositorio es una copia de un repositorio de otro usuario que para hacer cambios por tu cuenta o colaborar con el proyecto de otro usuario

• ¿Cómo crear un fork de un repositorio?

Desde la pagina del repositorio, se hace click en el botón fork y desde allí se le puede poner otro nombre y una descripción, luego se hace click en crear fork

• ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Se usa el comando git pull origin master “url de repositorio” si es la primera vez que lo solicitamos o si no lo subimos, sino solo git pull

• ¿Cómo aceptar una solicitud de extracción?

Desde la pagina del repositorio, hacer click en pull request, aparece una lista con todos los pull requests, seleccionar cualquiera y comparar diferencias, y hacer click en merge pull request

• ¿Qué es un etiqueta en Git?

Una etiqueta en git es similar a una branch, pero no modifica el codigo. Se puede usar para mas rapidamente saltar de version a version del repositorio

• ¿Cómo crear una etiqueta en Git?

Se usa el comando git tag “nombre de la etiqueta”

• ¿Cómo enviar una etiqueta a GitHub?

Se usa el comando git push –tags, esto es igual al git push pero tambien sube las etiquetas

• ¿Qué es un historial de Git?

Un historial de git muestra todos los cambios hechos en el repositorio, hasta antes de que empecemos a trabajar en el si lo clonamos desde un repositorio publico

• ¿Cómo ver el historial de Git?

Se usa el comando git log

• ¿Cómo buscar en el historial de Git?

Al usar git log, se pueden agregar filtros de días y horario, de cantidad de cambios, de estadísticas y muchas otras cosas mas para encontrar la información especifica que uno busca

• ¿Cómo borrar el historial de Git?

Se puede borrar la carpeta escondida Git local y luego volver a subir el codigo a un repositorio. Esto causa que no se pueda volver a una version anterior del repositorio

• ¿Qué es un repositorio privado en GitHub?

Un repositorio privado es un repositorio que no esta abierto a todos los usuarios, solo al creador y a los que este le permita el acceso, tambien pueden tener partes que estan abiertas para algunos usuarios y partes que no. Suelen ser usados por empresas

• ¿Cómo crear un repositorio privado en GitHub?

Al momento de crear un repositorio, se elige la opción de hacerlo privado

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Se puede invitar a alguien desde la pagina del repositorio entrando a “settings”, luego en “access” seleccionar “collaborators” y luego “add people”. Despues se usa un buscador para para escribir el nombre del usuario que se quiere invitar y le llegara un mail para poder acceder

• ¿Qué es un repositorio público en GitHub?

Un repositorio publico es uno al que pueden acceder todos los usuarios de github, por lo que pueden descargarlo, forkearlo y hacer cambios a este por su cuenta, o bien pueden colaborar con el tuyo y subir cambios que pueden luego ser confirmados por el creador

• ¿Cómo crear un repositorio público en GitHub?

Al momento de crear el repositorio, se elige la opción de que sea publico

• ¿Cómo compartir un repositorio público en GitHub?

Este puede ser accedido por todo el mundo, por lo que no tiene barreras para ser compartido

2) Realizar la siguiente actividad:

• Crear un repositorio.

o Dale un nombre al repositorio.
o Elije el repositorio sea público.
o Inicializa el repositorio con un archivo.

• Agregando un Archivo

o Crea un archivo simple, por ejemplo, "mi-archivo.txt".

o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
o Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

• Creando Branchs

o Crear una Branch
o Realizar cambios o agregar un archivo
o Subir la Branch

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

• Ve a GitHub e inicia sesión en tu cuenta.
• Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
• Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
• Opcionalmente, añade una descripción.
• Marca la opción "Initialize this repository with a README".
• Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

• Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).
• Abre la terminal o línea de comandos en tu máquina.
• Clona el repositorio usando el comando:
git clone https://github.com/tuusuario/conflict-exercise.git
• Entra en el directorio del repositorio:
cd conflict-exercise

Paso 3: Crear una nueva rama y editar un archivo
• Crea una nueva rama llamada feature-branch:
git checkout -b feature-branch
• Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:
Este es un cambio en la feature branch.
• Guarda los cambios y haz un commit:
git add README.md
git commit -m "Added a line in feature-branch"
Paso 4: Volver a la rama principal y editar el mismo archivo

• Cambia de vuelta a la rama principal (main):
git checkout main
• Edita el archivo README.md de nuevo, añadiendo una línea diferente:
Este es un cambio en la main branch.
• Guarda los cambios y haz un commit:
git add README.md
git commit -m "Added a line in main branch"

Paso 5: Hacer un merge y generar un conflicto

• Intenta hacer un merge de la feature-branch en la rama main:
git merge feature-branch
• Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

• Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:
<<<<<<< HEAD
Este es un cambio en la main branch.
=======
Este es un cambio en la feature branch.
>>>>>>> feature-branch

• Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.

• Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).

• Añade el archivo resuelto y completa el merge:
git add README.md
git commit -m "Resolved merge conflict"

Paso 7: Subir los cambios a GitHub

• Sube los cambios de la rama main al repositorio remoto en GitHub:
git push origin main
• También sube la feature-branch si deseas:
git push origin feature-branch

Paso 8: Verificar en GitHub
• Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
• Puedes revisar el historial de commits para ver el conflicto y su resolución.